

IN THE CLAIMS

Please amend the claims as follows:

1. (Currently Amended) A method for pipelining processing of a packet within a packet protocol handler, wherein inbound packet processing is performed within said packet protocol handler utilizing an inbound direct memory access (DMA) controller, said method comprising:

performing a packet processing task with respect to a packet in a first processor or a first thread, said performing a packet processing task comprising initiating a first processing thread within said first processor, wherein said first processing thread obtains a memory address for said packet and sets up an inbound DMA for said received packet utilizing said inbound DMA controller; and

performing subsequent packet processing tasks with respect to said packet utilizing subsequent processors or threads, wherein said subsequent processors or threads are dynamically allocated by:

utilizing said inbound DMA controller to move said packet from an inbound buffer into a memory location in accordance with said inbound DMA set up;

responsive to moving the header of said packet, issuing a header received synchronization signal from said inbound DMA controller to said available subsequent processor; and

responsive to receiving said header received synchronization signal, initiating a second processing thread within said subsequent available processor, wherein said second processing thread reads the packet header and fetches a control block in accordance with packet header information and processes said packet in accordance with said fetched control block.

2. (Original) The method of claim 1, wherein said subsequent processors or threads have associated input work queues, said method further comprising:

determining the available capacity of said input work queues; and

dynamically allocating said subsequent packet processing tasks to said subsequent processors or threads in accordance with said determined available capacity of said input work queues.

3. (Original) The method of claim 2, wherein said input work queue capacity availability determination includes:

determining whether or not one of said input work queues exceeds a predetermined threshold;

responsive to said one of said input work queues exceeding said predetermined threshold, determining the availability of an alternative one of said subsequent processors or threads; and

responsive to said one of said input work queues not exceeding said predetermined threshold, delivering a protocol processing task subset pointer into said one of said input work queues.

4. (Original) The method of claim 3, further comprising, responsive to said one of said input work queues not exceeding said predetermined threshold, issuing a pipeline synchronization signal to the subsequent processor or thread associated with said one of said input work queues.

5. (Currently Amended) The method of claim [[2]] 3, further comprising:

responsive to one of said input work queues not exceeding said predetermined threshold, placing a task pointer into said one of said input work queues.

Claims 6-7. (Cancelled)

8. (Currently Amended) The method of claim [[6]] 1, further comprising:

utilizing said inbound DMA controller to move said packet from an inbound DMA buffer into a memory location in accordance with said inbound DMA set up; and

responsive to moving said packet, placing the pointer to the memory location where said packet resides to the input work queue of said first processor or thread.

9. (Currently Amended) The method of claim [[7]] 1, wherein said processing said packet comprises:

responsive to determining that an acknowledgment signal is required:

generating an acknowledgment packet; and
issuing an acknowledgment synchronization signal to a third processing thread within a next available processor; and
responsive to receipt of said acknowledgment synchronization signal, performing an outbound direct memory access within said third processing thread.

10. (Currently Amended) A system for pipelining processing of a packet within a packet protocol handler, wherein inbound packet processing is performed within said packet protocol handler utilizing an inbound direct memory access (DMA) controller, said system comprising:

processing means for performing a packet processing task with respect to a packet in a first processor ~~or a first thread~~, said processing means for performing a packet processing task comprising processing means for initiating a first processing thread within said first processor, wherein said first processing thread obtains a memory address for said packet and sets up an inbound DMA for said received packet utilizing said inbound DMA controller; and

processing means for performing subsequent packet processing tasks with respect to said packet utilizing subsequent processors or threads, wherein said subsequent processors or threads are dynamically allocated by:

processing means within said inbound DMA controller for moving said packet from an inbound buffer into a memory location in accordance with said inbound DMA set up;

processing means responsive to moving the header of said packet for issuing a header received synchronization signal from said inbound DMA controller to said available subsequent processor; and

processing means responsive to receiving said header received synchronization signal for initiating a second processing thread within said subsequent available processor, wherein said second processing thread reads the packet header and fetches a control block in accordance with packet header information and processes said packet in accordance with said fetched control block.

11. (Original) The system of claim 10, wherein said subsequent processors or threads have associated input work queues, said system further comprising:

processing means for determining the available capacity of said input work queues; and

processing means for dynamically allocating said subsequent packet processing tasks to said subsequent processors or threads in accordance with said determined available capacity of said input work queues.

12. (Original) The system of claim 11, wherein said processing means for determining the available capacity of said input work queue includes:

processing means for determining whether or not one of said input work queues exceeds a predetermined threshold;

processing means responsive to said one of said input work queues exceeding said predetermined threshold for determining the availability of an alternative one of said subsequent processors or threads; and

processing means responsive to said one of said input work queues not exceeding said predetermined threshold for delivering a protocol processing task subset pointer into said one of said input work queues.

13. (Original) The system of claim 12, further comprising, processing means responsive to said one of said input work queues not exceeding said predetermined threshold for issuing a pipeline synchronization signal to the subsequent processor or thread associated with said one of said input work queues.

14. (Currently Amended) The system of claim ~~11~~ 12, further comprising:

processing means responsive to one of said input work queues not exceeding said predetermined threshold for placing a task pointer into said one of said input work queues.

Claims 15-16. (Cancelled)

17. (Currently Amended) The system of claim ~~15~~ 10, further comprising:

processing means within said inbound DMA controller for moving said packet from an inbound DMA buffer into a memory location in accordance with said inbound DMA set up; and

processing means responsive to moving said packet for placing the pointer to the memory location where said packet resides to the input work queue of said first processor or thread.

18. (Currently Amended) The system of claim ~~46~~ 10, wherein said processing means for processing said packet comprises:

processing means responsive to determining that an acknowledgment signal is required for:

generating an acknowledgment packet; and

issuing an acknowledgment synchronization signal to a third processing thread within a next available processor; and

processing means responsive to receipt of said acknowledgment synchronization signal for performing an outbound direct memory access within said third processing thread.